

# DYSON–SCHWINGER EQUATIONS IN THE THEORY OF COMPUTATION

COLLEEN DELANEY AND MATILDE MARCOLLI

**ABSTRACT.** Following Manin’s approach to renormalization in the theory of computation, we investigate Dyson–Schwinger equations on Hopf algebras, operads and properads of flow charts, as a way of encoding self-similarity structures in the theory of algorithms computing primitive and partial recursive functions and in the halting problem.

## 1. INTRODUCTION

There are many deep connections between theoretical physics and information theory, and in particular the theory of computation, see for instance the account given in [1]. In the recent papers [21] and [22], Manin developed a new approach to the theory of computation and the halting problem, based on importing ideas and techniques from the Hopf-algebraic formulation of renormalization in perturbative quantum field theory, [3], [9], [15]. The purpose of this paper is to show that the Hopf algebra of flow charts introduced by Manin in [21] exhibits self-similarity structures given by solutions of combinatorial Dyson–Schwinger equations, defined as in perturbative renormalization, [2], [10], [11], [12], [19], [30]. These can be thought of as a notion of “equations of motion” in the theory of computation.

Dyson–Schwinger equations [7], [26] are a formulation of equations of motion in perturbative quantum field theory, expressed in the form of relations between Green functions. In recent years, an algebraic formulation of the combinatorial structure of perturbative renormalization for scalar field theories was developed, starting with the work of Kreimer on the Hopf-algebraic structure of renormalization [15], followed by the Connes–Kreimer formulation [3] of the BPHZ renormalization procedure, and the formulation in terms of a Riemann–Hilbert correspondence for categories of differential systems in Connes–Marcolli [5], [6]. The Connes–Kreimer formulation of renormalization was given a very general algebraic formulation in terms of Rota–Baxter algebras in the work of Ebrahimi-Fard, Guo and Kreimer [9]. Correspondingly, the Dyson–Schwinger equations were also given a combinatorial form, reflecting the Hopf-algebraic structure of renormalization, in the work of Bergbauer, Kreimer, and Yeats, [2], [19], [30] and more recently with an extensive study by Foissy, [10], [11], [12].

In this paper we investigate the formulation of Dyson–Schwinger equations in the context of Manin’s approach to renormalization and computation. In §2 we review some variants of the construction of a Hopf algebra of *flow charts*. These are diagrams, consisting of decorated planar rooted trees, that compute primitive recursive functions. In §3 we discuss grafting operators and Dyson–Schwinger equations for flow charts, as a way of identifying certain self-similarity structures in the computation of primitive recursive functions. We review some known results about existence and uniqueness of solutions and on conditions under which the coordinates of the solution generate a Hopf subalgebra. We show that one can also consider ideals generated by the coordinates of solutions. Under the condition that the grafting operator is a cocycle, these can give rise to Hopf

ideals. The quotient Hopf algebra determines a subgroup of the affine group scheme dual to the (commutative) Hopf algebra of flow charts. This subgroup defines a Galois group in the sense of Yanofsky's Galois theory of algorithm [29], consisting of those symmetries that are compatible with the self-similarity structure imposed by the Dyson–Schwinger equation. We then show an explicit example of a Dyson–Schwinger equation based on computation by binary trees that does not satisfy the Hopf algebra condition. In order to accommodate this type of example, that is relevant to the setting of computation, we show that it is convenient to take an operadic viewpoint on Dyson–Schwinger equations, which was already suggested in the work of Bergbauer–Kreimer [2]. In §4 we show that Dyson–Schwinger equations can be formulated in the operad setting, using a family of grafting operations, which is not required to satisfy a cocycle condition. We consider Dyson–Schwinger equations on an *operad of flow charts*. We prove the existence and uniqueness of solutions for these operadic Dyson–Schwinger equations and also for a *properad* version. The extension from operads to properads is motivated by the more recent formulation, given by Manin in [24], of a *properad of flow charts* based on directed (acyclic) graphs instead of rooted trees. Finally, in §5 we discuss the algebraic Feynman rule associated to the halting problem, as in [22] and its BPHZ renormalization and the role of Dyson–Schwinger equations in the halting problem.

## 2. PRIMITIVE RECURSIVE FUNCTIONS AND THE HOPF ALGEBRA OF FLOW CHARTS

We recall here the setup of Manin [21] on the Hopf algebra of flow charts. First recall from [20] the following facts about primitive recursive functions.

**2.1. Primitive recursive functions.** As in §V.2 of [20], we consider the class of *primitive recursive functions* as generated by the *basic functions*

- Successor  $s : \mathbb{N} \rightarrow \mathbb{N}$ ,  $s(x) = x + 1$ ;
- Constant  $c^n : \mathbb{N}^n \rightarrow \mathbb{N}$ ,  $c^n(x) = 1$  (for  $n \geq 0$ );
- Projection  $\pi_i^n : \mathbb{N}^n \rightarrow \mathbb{N}$ ,  $\pi_i^n(x) = x_i$  (for  $n \geq 1$ );

With the *elementary operations*

- Composition (substitution)  $\mathfrak{c}_{(m,m,p)}$ : for  $f : \mathbb{N}^m \rightarrow \mathbb{N}^n$ ,  $g : \mathbb{N}^n \rightarrow \mathbb{N}^p$ ,

$$g \circ f : \mathbb{N}^m \rightarrow \mathbb{N}^p, \quad \mathcal{D}(g \circ f) = f^{-1}(\mathcal{D}(g));$$

- Bracketing (juxtaposition)  $\mathfrak{b}_{(k,m,n_i)}$ : for  $f_i : \mathbb{N}^m \rightarrow \mathbb{N}^{n_i}$ ,  $i = 1, \dots, k$ ,

$$f = (f_1, \dots, f_k) : \mathbb{N}^m \rightarrow \mathbb{N}^{n_1 + \dots + n_k}, \quad \mathcal{D}(f) = \mathcal{D}(f_1) \cap \dots \cap \mathcal{D}(f_k);$$

- Recursion  $\mathfrak{r}_n$ : for  $f : \mathbb{N}^n \rightarrow \mathbb{N}$  and  $g : \mathbb{N}^{n+2} \rightarrow \mathbb{N}$ ,

$$h(x_1, \dots, x_n, 1) := f(x_1, \dots, x_n),$$

$$h(x_1, \dots, x_n, k+1) := g(x_1, \dots, x_n, k, h(x_1, \dots, x_n, k)), \quad k \geq 1,$$

where recursively  $(x_1, \dots, x_n, 1) \in \mathcal{D}(h)$  iff  $(x_1, \dots, x_n) \in \mathcal{D}(f)$  and  $(x_1, \dots, x_n, k+1) \in \mathcal{D}(h)$  iff  $(x_1, \dots, x_n, k, h(x_1, \dots, x_n, k)) \in \mathcal{D}(g)$ .

**2.2. Hopf algebras of decorated rooted trees.** We recall here the construction of Hopf algebras of decorated rooted trees and of planar decorated rooted trees and their main properties, see [11] and references therein. Throughout this paper vector spaces and algebras are over a field  $\mathbb{K}$  of characteristic zero.

**2.2.1. Rooted trees.** A rooted tree is a finite graph  $\tau$ , whose geometric realization  $|\tau|$  is simply connected, defined by combinatorial data  $(F_\tau, V_\tau, v_\tau, \delta_\tau, j_\tau)$ , with  $F_\tau$  a set of half-edges (also called flags or tails),  $V_\tau$  the set of vertices with a distinguished element  $v_\tau \in V_\tau$  (the root), a boundary map  $\partial_\tau : F_\tau \rightarrow V_\tau$  that associates to half-edge its boundary vertex and an involution  $j_\tau : F_\tau \rightarrow F_\tau$ ,  $j_\tau^2 = 1$  that performs the matching of half-edges that determines the edges of  $\tau$ . The resulting graph has edges (or internal edges) given by pairs of half-edges matched by the involution, and tails (or external edges) given by half-edges that are fixed by the involution. We denote by  $E_\tau$  the set of internal edges and by  $E_\tau^{ext}$  the set of external edges.

**2.2.2. Orientations.** We consider rooted trees as endowed with the orientation that the root vertex as the *output*, namely where all edges are oriented in the direction of the unique path to the root.

**2.2.3. External edges.** We also assume that the root vertex has an outgoing half-edge and a number of incoming edges, while all other vertices have one outgoing edge and a number of incoming edges and possibly a number of incoming tails. Each leaf of the tree has an incoming tail.

**2.2.4. Planarity.** A *planar* rooted tree is a rooted tree  $\tau$  together with a fixed embedding  $\iota : |\tau| \hookrightarrow \mathbb{R}^2$  of its geometric realization in the plane.

**2.2.5. Decorations.** A (planar) decorated rooted tree is rooted tree and with a map  $\phi_V : V_\tau \rightarrow \mathcal{D}_V$  of the set of vertices to a set  $\mathcal{D}_V$  of vertex-labels and a map  $\phi_F : F_\tau \rightarrow \mathcal{D}_F$  to a set  $\mathcal{D}_F$  of labels of flags. The assignment of a decoration  $(\phi_V, \phi_F)$  to a rooted tree  $\tau$  is in general subject to constraints. For example, if  $f$  and  $f'$  in  $F_\tau$  are matched by the involution  $j_\tau$  to form an edge  $e \in E_\tau$ , then they should carry the same decoration  $\phi_F(f) = \phi_F(f')$ . This defines a labeling of edges. We will see other constraints below, in the specific case of the Hopf algebra of flow charts. We also consider the forgetful map that forgets the flag labels  $\mathcal{D}_F$  and retains the vertex labels  $\mathcal{D}_V$  and a further forgetful map to unlabeled trees.

**2.2.6. Noncommutative Hopf algebra of planar rooted trees.** Given a (planar) rooted tree  $\tau$ , an *admissible cut*  $C$  of  $\tau$  is a modification of the involution  $j_\tau$  that *cuts* a subset of internal edges  $e_i \in E_\tau$  into two flags  $f_i, f'_i$ . Namely, instead of having  $j_\tau(f_i) = f'_i$ , one modifies the involution so that the new  $\hat{j}(f_i) = f_i$  and  $\hat{j}(f'_i) = f'_i$ , where the set of edges  $e_i$  is chosen in such a way that every oriented path in  $\tau$  from a leaf vertex to the root contains at most one  $e_i$ . The new graph obtained using the involution  $\hat{j}$  is a forest

$$C(\tau) = \rho_C(\tau) \amalg \pi_C(\tau)$$

consisting of one (planar) rooted tree  $\rho_C(\tau)$  containing the root vertex of  $\tau$  and a finite disjoint union of other oriented planar trees  $\pi_C(\tau) = \amalg_i \pi_{C,i}(\tau)$ , where each tree  $\pi_{C,i}(\tau)$  has a single output external edge, to which we assign as root the boundary vertex of this output.

The *noncommutative* Hopf algebra of planar rooted trees  $\mathcal{H}^{nc}$  is defined, as an algebra, as the free algebra generated by the planar rooted trees, with the coalgebra structure given by the *admissible-cuts* coproduct

$$(2.1) \quad \Delta(\tau) = \tau \otimes 1 + 1 \otimes \tau + \sum_C \pi_C(\tau) \otimes \rho_C(\tau).$$

The coproduct is coassociative and the Hopf algebra is graded by the number of vertices of rooted trees, hence the antipode is defined inductively by the formula

$$S(x) = -x - \sum S(x')x'', \quad \text{for } \Delta(x) = x \otimes 1 + 1 \otimes x + \sum x' \otimes x''$$

with  $x'$ ,  $x''$  terms of lower degree. See [13], [14] for more details on this Hopf algebra. An element  $x$  in a Hopf algebra is primitive if  $\Delta(x) = 1 \otimes x + x \otimes 1$ .

**2.2.7. Commutative Hopf algebra of rooted trees.** In the Hopf algebraic approach to perturbative renormalization, as formulated by Kreimer [15] and Connes–Kreimer [3], one considers a *commutative* Hopf algebra of rooted trees decorated by Feynman graphs, with the admissible-cuts coproduct (2.1), or else a commutative Hopf algebra of Feynman graphs, with a coproduct that corresponds to inclusions of subgraphs (subdivergences) and quotient graphs.

In the case of planar rooted trees, one can also consider a commutative Hopf algebra  $\mathcal{H}$ , which is a quotient of  $\mathcal{H}^{nc}$  by imposing the commutativity of multiplication. The generators of  $\mathcal{H}$  are still planar rooted trees, but monomials in these generators no longer identify with *embedded* forests. The grading and antipode are as above, induced by those of  $\mathcal{H}^{nc}$ .

**2.3. Manin’s Hopf algebra of flow charts.** We consider, as in [21], the set of planar labelled rooted trees, where the label set  $\mathcal{D}_V$  of vertices is given by the set of elementary operations

$$\{\mathbf{c}_{(m,n,p)}, \mathbf{b}_{(k,m,n_i)}, \mathbf{r}_n\}$$

(composition, bracketing, recursion) and the label set  $\mathcal{D}_F$  of the flags is the set of primitive recursive functions. Notice that, because the inputs of the composition, bracketing and recursion operations are ordered, we need to work with *planar* rooted trees.

The data  $(\phi_V(v), \phi_F(f_i))$  of a labelings  $\phi_V(v) \in \mathcal{D}_V$  and  $\phi_F(f_i)$  of a vertex  $v \in V_\tau$  and of all the half-edges  $f_i \in F_\tau$  with  $\partial(f_i) = v$  are *admissible* if they satisfy the following conditions:

- If  $\phi_V(v) = \mathbf{c}_{(m,n,p)}$ , then  $v$  must have valence three; the labels  $h_1 = \phi_F(f_1)$  and  $h_2 = \phi_F(f_2)$  of the two incoming flags must have domains and ranges  $h_1 : \mathbb{N}^m \rightarrow \mathbb{N}^n$  and  $h_2 : \mathbb{N}^n \rightarrow \mathbb{N}^p$  and the outgoing flag must be labeled by the composition  $h_2 \circ h_1 = \mathbf{c}_{(m,n,p)}(h_1, h_2)$ .
- If  $\phi_V(v) = \mathbf{r}_n$ , then  $v$  must have valence three; the labels  $h_1 = \phi_F(f_1)$  and  $h_2 = \phi_F(f_2)$  of the two incoming flags must have domains and ranges  $h_1 : \mathbb{N}^n \rightarrow \mathbb{N}$  and  $h_2 : \mathbb{N}^{n+2} \rightarrow \mathbb{N}$  and the outgoing flag must be labeled by the recursion  $h = \mathbf{r}_n(h_1, h_2)$ .
- If  $\phi_V(v) = \mathbf{b}_{(k,m,n_i)}$  then  $v$  must have valence  $k+1$  and all the functions  $h_i = \phi_F(f_i)$  associated to the incoming flags must have domain in the same  $\mathbb{N}^m$ . The outgoing flag must be labeled by the bracketing  $f = (f_1, \dots, f_k) = \mathbf{b}_{(k,m,n_i)}(f_1, \dots, f_k)$ .

**Definition 2.1.** An admissible labeling of a planar rooted tree  $\tau$  by primitive recursive functions is a collection of labelings

$$\{(\phi_V(v), \phi_F(f_i)) \mid v \in V_\tau \text{ and } f_i \in F_\tau \text{ with } \partial(f_i) = v\}$$

such that the admissibility condition above is satisfied at all vertices and such that  $\phi_F(f) = \phi_F(f')$  whenever  $j_\tau(f) = f'$ .

A vertex labeling of a planar rooted tree  $\tau$  by elementary operations is a collection of labeling  $\{\phi_V(v) \in \{\mathbf{c}, \mathbf{r}, \mathbf{b}_k\} \mid v \in V_\tau\}$  with the only constraint that labels of type  $\mathbf{c}$  and  $\mathbf{r}$  can only be assigned to vertices with two incoming flags and one outgoing flag and labels of type  $\mathbf{b}_k$  can be assigned to vertices with  $k$  incoming flags and one outgoing flag.

Via the forgetful map that keeps the vertex labels and forgets the flag labels, a planar rooted tree with an admissible labeling by primitive recursive functions determines one with vertex labeling by elementary operations, while not all vertex labeling by elementary operations on a given planar rooted tree will admit a compatible flag labeling. Notice that, in the case of vertex labelings, we no longer distinguish between labels  $\mathbf{c}_{(m,n,p)}$ , with different choices of  $m$ ,  $n$  and  $p$ , since we do not fix

the flag labels: all these vertex labels in  $\mathcal{D}_V$  correspond to the same label  $\mathbf{c}$ . Similarly, all the labels  $\mathbf{r}_n$  of admissible labeling correspond to the same label  $\mathbf{r}$  of vertex labeling, and all the  $\mathbf{b}_{(k,m,n_i)}$  correspond to the same  $\mathbf{b}_k$ . We retain the different  $\mathbf{b}_k$  as vertex labelings as those correspond to different valences of the vertex.

Thus, a planar rooted tree with an admissible labeling by primitive recursive functions can be interpreted as a *flow chart* that computes the output function starting from the functions associated to the incoming external edges according to the operations performed at the vertices.

A planar rooted tree  $\tau$  with a vertex labeling by elementary operations, on the other hand, should be thought of as a template of a possible computational scheme: when applied to a tuple of primitive recursive functions  $(h_1, \dots, h_k)$ , with  $k$  the number of incoming flags of  $\tau$ , it either computes an output functions  $h$ , if the inputs  $(h_1, \dots, h_k)$  together with the assigned vertex labels determine an admissible labeling of  $\tau$  by partial recursive function, or else we set its output to be the empty function.

We then define two slightly different versions of the Hopf algebra of flow charts, depending on the type of labeling that we want to use on trees.

**Definition 2.2.** *The noncommutative Hopf algebra  $\mathcal{H}_{\text{flow}, \mathcal{P}}^{\text{nc}}$  of flow charts is the free algebra generated by the planar rooted trees with admissible labelings by primitive recursive functions, with the coproduct (2.1). We also denote by  $\mathcal{H}_{\text{flow}, \mathcal{P}}$  the commutative quotient of  $\mathcal{H}_{\text{flow}, \mathcal{P}}^{\text{nc}}$ . Similarly, the noncommutative Hopf algebra  $\mathcal{H}_{\text{flow}, \mathcal{V}}^{\text{nc}}$  is the free algebra generated by planar rooted trees with vertex labeling by elementary operations, again with the admissible-cuts coproduct. We denote by  $\mathcal{H}_{\text{flow}, \mathcal{V}}$  its commutative quotient.*

Notice that if  $\tau$  has an admissible labeling, then the rooted trees  $\rho_C(\tau)$  and  $\pi_{C,i}(\tau)$  also inherit induced admissible labelings, so the Hopf algebras are well defined. There is a Hopf algebra homomorphism  $\mathcal{H}_{\text{flow}, \mathcal{P}}^{\text{nc}} \rightarrow \mathcal{H}_{\text{flow}, \mathcal{V}}$ , determined by forgetting the flag labels, but as observed above not all generators of  $\mathcal{H}_{\text{flow}, \mathcal{V}}$  are in the image of this map. Since the rooted trees in  $\mathcal{H}_{\text{flow}, \mathcal{V}}$  have no labels attached to flags, we can represent them as rooted trees without external edges. This will be implicitly done in the description of the grafting operators in Lemma 3.1 below.

In the following, whenever a statement applies to both types of tree decorations on the Hopf algebras of flow charts, we will drop the  $\mathcal{P}$  and  $\mathcal{V}$  subscripts.

**2.3.1. Binary versus  $k$ -ary operations.** It is possible to reformulate the Hopf algebra of flow charts by using only *binary* trees. In fact, it is shown in §2 of [28] that the description of primitive recursive functions in terms of basic functions (successor, constants and projections) and elementary operations (composition, bracketing and recursion), as recalled above, can be reformulated as repeated applications of *binary* operations of these same forms. In fact, the composition and recursion operations already are binary (they label vertices with two input flags and one output) so one only needs to verify that the bracketing operation  $\mathbf{b}_{(k,m,n_i)}$  assigning to  $k$  functions  $f_i : \mathbb{N}^m \rightarrow \mathbb{N}^{n_i}$ ,  $i = 1, \dots, k$  their juxtaposition  $f = (f_1, \dots, f_k) : \mathbb{N}^m \rightarrow \mathbb{N}^{n_1 + \dots + n_k}$  is a composition

$$\mathbf{b}_{(k,m,n_i)} = \mathbf{b}_{(2,m,n_1,n_2+\dots+n_k)} \circ \dots \circ \mathbf{b}_{(2,m,n_{k-1},n_k)}.$$

This replaces a single vertex with  $k$  inputs and one output with  $k$  vertices with two inputs and one output. Thus, without loss of generality in the algorithmic representation of primitive recursive functions, one can consider a sub-Hopf algebra  $\mathcal{H}_{\text{flow}}^{\text{nc},0} \subset \mathcal{H}_{\text{flow}}^{\text{nc}}$  generated by *binary* planar rooted tree with an admissible labeling by primitive recursive functions, and the corresponding commutative quotient  $\mathcal{H}_{\text{flow}}^0 \subset \mathcal{H}_{\text{flow}}$ .

One the other hand, one can also extend the two binary operations of composition and recursion to  $k$ -ary operations for arbitrary  $k \geq 2$ . In the case of composition, we can introduce a new set of vertex labels  $\mathfrak{c}_{(k,n_i)}$  that correspond to the  $k$ -ary compositions  $\mathfrak{c}_{(k,m,n_i)}(h_i) = h_k \circ \dots \circ h_1$  of functions  $h_i : \mathbb{N}^{n_{i-1}} \rightarrow \mathbb{N}^{n_i}$ , for  $i = 1, \dots, k$ , with  $n_0 = m$ . In the case of recursion, for  $k \geq 1$ , one can consider  $(k+1)$ -ary recursions where the recursion depends on  $k$  initial conditions. Namely, we define  $\mathfrak{r}_{k+1,n}$  as the operation that computes the function  $h = \mathfrak{r}_{k,n}(h_1, \dots, h_{k+1}) : \mathbb{N}^{n+1} \rightarrow \mathbb{N}$ , from the input  $h_i : \mathbb{N}^n \rightarrow \mathbb{N}$  for  $i = 1, \dots, k$  and  $h_{k+1} : \mathbb{N}^{2n+1} \rightarrow \mathbb{N}$ , as

$$(2.2) \quad \begin{aligned} h(x_1, \dots, x_n, 1) &= h_1(x_1, \dots, x_n), \\ &\vdots \\ h(x_1, \dots, x_n, k) &= h_k(x_1, \dots, x_n), \\ h(x_1, \dots, x_n, k + \ell) &= h_{k+1}(x_1, \dots, x_n, h_1(x_1, \dots, x_n), \dots, h_k(x_1, \dots, x_n), k + \ell - 1), \\ &\text{for } \ell \geq 1 \end{aligned}$$

These  $k$ -ary operations are clearly reducible to compositions of binary operations, hence any primitive recursive function that can be computed using these operations can also be computed by the presentation in terms of binary compositions and recursions. However, as we shall discuss more in detail below, it is convenient to include these explicitly in the structure of the Hopf algebra of flow charts in order to have well defined grafting operators that give rise to Hochschild cocycles and to Dyson–Schwinger equations.

Thus, we propose another minor modification of Manin’s Hopf algebra of flow charts, as follows.

**Definition 2.3.** We denote by  $\mathcal{H}_{\text{flow}, \mathcal{P}'}^{nc}$  and  $\mathcal{H}_{\text{flow}, \mathcal{V}'}^{nc}$  the noncommutative Hopf algebras of flow charts where the vertex label sets is enlarged to contain all the  $k$ -ary compositions and recursions  $\mathfrak{c}_{(k,n_i)}$  and  $\mathfrak{r}_{k,n}$ , and the admissibility condition is restated accordingly. We denote by  $\mathcal{H}_{\text{flow}, \mathcal{P}'}$  and  $\mathcal{H}_{\text{flow}, \mathcal{V}'}$  their commutative quotients.

### 3. DYSON–SCHWINGER EQUATIONS IN THE HOPF ALGEBRA OF FLOW CHARTS

Recall here the formulation of combinatorial Dyson–Schwinger equations in Hopf algebras of decorated rooted trees, following [2], [30] and especially [10], [11], [12]. We then focus on the specific case of the Hopf algebras of flow charts described above.

**3.1. Insertion operators for decorated planar rooted trees.** We work here with the Hopf algebra of flow charts  $\mathcal{H}_{\text{flow}, \mathcal{V}'}^{nc}$ . By analogy with the setting of perturbative renormalization [2], [30], we define, for each type of elementary operation  $\delta \in \{\mathfrak{b}, \mathfrak{c}, \mathfrak{r}\}$ , a grafting operator  $B_\delta^+$  defined in the following way. Given a monomial  $T$  in  $\mathcal{H}_{\text{flow}, \mathcal{V}'}^{nc}$ , which consists of a forest with vertex labeling by elementary operations, we define  $B_\delta^+(T)$  as the sum of planar graphs obtained by adding a new root vertex with a number of incoming flags equal to the number of trees in  $T$  and a single output flag, with the vertex decorated by  $\delta$ .

**3.1.1. Hochschild 1-cocycles.** In the case of perturbative renormalization, it is well known [2] that the analogous grafting operators  $B_\delta^+$ , with vertex decorations  $\delta$ , define Hochschild 1-cocycles.

The 1-cocycle condition for the operator  $B_\delta^+$  then consists of the property that

$$(3.1) \quad \Delta B_\delta^+ = (id \otimes B_\delta^+) \Delta + B_\delta^+ \otimes 1.$$

**Lemma 3.1.** The operators  $B_\delta^+$  on  $\mathcal{H}_{\text{flow}, \mathcal{V}'}^{nc}$ , with  $\delta \in \{\mathfrak{b}, \mathfrak{c}, \mathfrak{r}\}$ , satisfy the 1-cocycle condition (3.1).

*Proof.* This case works exactly as in the renormalization setting, see [2]: we reproduce the argument here for the reader's convenience. For an element  $x$  in a Hopf algebra  $\mathcal{H}$  with coproduct

$$\Delta(x) = 1 \otimes x + x \otimes 1 + \sum x' \otimes x'',$$

where  $x'$  and  $x''$  denote lower degree terms, we use the notation

$$\tilde{\Delta}(x) := \sum x' \otimes x''.$$

One then sees easily that the cocycle condition (3.1) is equivalent to

$$(3.2) \quad \tilde{\Delta}B_\delta^+ = (id \otimes B_\delta^+)\tilde{\Delta} + id \otimes B_\delta^+(1),$$

where, as in the renormalization case, with  $B_\delta^+(1) = v_\delta$  is a single vertex with assigned label  $\delta$ . The argument is then as in the original case, with the first term in the right hand side of (3.2) accounting for the admissible cuts where the root vertex remains attached to the  $\rho_C(T)$  part of an admissible cut of  $T$ , and the second term counts the case where the admissible cut separates the root vertex completely.  $\square$

Notice that, for the cocycle condition to hold, we need to be able to assign the same type of label (**b**, **c**, or **r**) to vertices of arbitrary valence, hence the reason for including in the set of vertex decorations all the  $k$ -ary versions of composition and recursion and working with the version  $\mathcal{H}_{\text{flow}, \mathcal{V}}^{nc}$  of the Hopf algebra of flow charts.

It is possible to define other interesting grafting operators in the Hopf algebras of flow charts, which, however, do not satisfy the cocycle condition. These can still be used to construct Dyson–Schwinger equations, but the set of solutions does not define a Hopf subalgebra. This will be discussed in an example in the following subsection.

**3.2. Systems of combinatorial Dyson–Schwinger equations.** Given a graded Hopf algebra  $\mathcal{H}$ , we consider the direct product  $\overline{\mathcal{H}} = \prod_{n=0}^{\infty} \mathcal{H}_n$ , whose elements we write as infinite sums  $x = \sum_{n=0}^{\infty} x_n$ , with  $x_n \in \mathcal{H}_n$ , endowed with the associative product and coproduct induced by those of  $\mathcal{H}$ . Following [10], (see also [2], [30]) given a formal power series

$$P(t) = \sum_{k=0}^{\infty} a_k t^k, \quad \text{with } a_0 = 1,$$

and a Hochschild 1-cocycle  $B^+$  on the Hopf algebra  $\mathcal{H}$ , the associated Dyson–Schwinger equation is given by

$$(3.3) \quad X = B^+(P(X)).$$

One considers this as an equation in  $\overline{\mathcal{H}}$  and interprets the infinite sum  $P(X) = \sum_k a_k X^k$  as an element in  $\overline{\mathcal{H}}$ .

One should read (3.3) as a fixed point equation for the nonlinear transformation

$$X \mapsto B^+(P(X))$$

of  $\overline{\mathcal{H}}$ . Solutions to the fixed point equation are elements of  $\overline{\mathcal{H}}$  that exhibit a self-similarity property with respect to the operation of first applying  $X \mapsto P(X)$  (in the case of a polynomial, this operation heuristically replaces  $X$  by a sum of multiple copies of itself weighted by the coefficients of the polynomial) and then grafting them together according to the cocycle  $B^+$ .

In fact, the equation (3.3) has a unique solution given by an element  $X = \sum_k x_k$  with  $x_k \in \mathcal{H}_k$  whose homogeneous components are determined inductively by the procedure

$$(3.4) \quad x_{n+1} = \sum_{k=1}^n \sum_{j_1 + \dots + j_k = n} a_k B^+(x_{j_1} \cdots x_{j_k}),$$

starting with  $x_1$  given by  $B^+(1)$ , see Lemma 2 of [2] and Proposition 2 of [10].

Notice that we can still define the Dyson–Schwinger equation (3.3) for other types of grafting operators that do not necessarily satisfy the 1-cocycle condition (3.1), such as the examples discussed in the previous subsection. The existence and uniqueness of solutions, of the form (3.4) is also still valid. Where the 1-cocycle condition is crucially used is in showing that the associative subalgebra of  $\mathcal{H}$  generated by the components  $x_k$  of the solution is also a Hopf subalgebra, see Theorem 3 of [2].

**3.2.1. Systems of Dyson–Schwinger equations.** As discussed in the work of Foissy [11], in cases like our  $\mathcal{H}_{\text{flow}, \mathcal{V}'}^{nc}$ , where one has different vertex labels  $\delta \in \{\mathfrak{b}, \mathfrak{c}, \mathfrak{r}\}$ , one can consider more complicated *systems* of Dyson–Schwinger equations, involving the grafting operators  $B_\delta^+$  on  $\mathcal{H}_{\text{flow}, \mathcal{V}'}^{nc}$ .

For  $\delta \in \{\mathfrak{b}, \mathfrak{c}, \mathfrak{r}\}$ , we consider the grafting cocycles  $B_\delta^+$  on the Hopf algebra  $\mathcal{H}_{\text{flow}, \mathcal{V}'}^{nc}$ , as above. We also consider data of three non-constant formal power series in three variables  $X = (X_\delta)$

$$F_\delta(X) = \sum_{k_1, k_2, k_3} a_{k_1, k_2, k_3}^{(\delta)} X_{\mathfrak{b}}^{k_1} X_{\mathfrak{c}}^{k_2} X_{\mathfrak{r}}^{k_3}.$$

To these data we associate the system of Dyson–Schwinger equations as in [11]

$$(3.5) \quad X_\delta = B_\delta^+(F_\delta(X)).$$

As shown in Proposition 5 of [11], these systems of equations also have a unique solution of the form  $X_\delta = \sum_\tau x_\tau \tau$ , with the sum over all planar rooted trees with root decorated by  $\delta$ , with coefficients

$$x_\tau = \left( \prod_{k=1}^3 \frac{(\sum_{l=1}^{m_k} p_{\delta, l})!}{\prod_{l=1}^{m_k} p_{\delta, l}!} \right) a_{\sum_{k=1}^3 p_{1, k}, \sum_{k=1}^3 p_{2, k}, \sum_{k=1}^3 p_{3, k}}^{(\delta)} x_{\tau_{1,1}}^{p_{1,1}} \cdots x_{\tau_{3, m_3}}^{p_{3, m_3}},$$

when

$$\tau = B^+(\tau_{1,1}^{p_{1,1}} \cdots \tau_{1, m_1}^{p_{1, m_1}} \cdots \tau_{3,1}^{p_{3,1}} \cdots \tau_{3, m_3}^{p_{3, m_3}}).$$

**3.3. Hopf subalgebras and Hopf ideals.** Bergbauer and Kreimer showed that, under the assumption that  $B_\delta^+$  is a Hochschild cocycle, solutions of Dyson–Schwinger equations determine a sub-Hopf algebra. Namely, they considered Dyson–Schwinger equation in a Hopf algebra  $\mathcal{H}$  of the form

$$(3.6) \quad X = 1 + \sum_{n=1}^{\infty} c_n B_\delta^+(X^{n+1}).$$

One then considers the associative algebra  $\mathcal{A}$  defined as the *subalgebra* of  $\mathcal{H}$  generated by the components  $x_n$ , with  $n \geq 0$  of the unique solution of this Dyson–Schwinger equations. In Theorem 3 of [2], they showed that  $\mathcal{A}$  is in fact a *sub-Hopf* algebra, by inductively using the cocycle condition (3.1) to see that

$$(3.7) \quad \Delta(x_n) = \sum_{k=0}^n \Pi_k^n \otimes x_k, \quad \text{where} \quad \Pi_k^n = \sum_{j_1 + \dots + j_{k+1} = n-k} x_{j_1} \cdots x_{j_{k+1}}.$$



This result was extended by Foissy to the more general form (3.3) of Dyson–Schwinger equations in [10] and to systems of Dyson–Schwinger equations in [11]. In Theorem 4 of [10] it is shown that, for an equation of the form (3.3), the subalgebra  $\mathcal{A}$  spanned by the solutions is a Hopf subalgebra if and only if the formal series  $P(t)$  satisfies the differential equation

$$(3.8) \quad (1 - \alpha\beta t)P'(t) = \alpha P(t), \quad \text{with} \quad P(0) = 1,$$

for some  $\alpha, \beta \in \mathbb{K}$ , again under the assumption that  $B^+$  satisfies the cocycle condition (3.1). Similarly, in [11] combinatorial conditions and conditions on the multivariable series  $F_i$  are identified that completely characterize when the solutions of that systems of Dyson–Schwinger equations generate a Hopf algebra, under the assumption that the  $B^+$  satisfies the cocycle condition (3.1).

**3.3.1. Ideals of Dyson–Schwinger solutions.** It is natural to consider not only the associative *subalgebra*  $\mathcal{A}$  spanned by the components of the solutions of Dyson–Schwinger equations, but also the *ideal*  $\mathcal{I}$  spanned by the  $(x_n)_{n \geq 1}$ . As in the case of the subalgebra  $\mathcal{A}$ , it is also natural to ask for conditions that will ensure that it is a Hopf subalgebra, so with the ideal  $\mathcal{I}$  of solutions it is natural to ask for conditions ensuring that it is a Hopf ideal, so that it makes sense to define a quotient Hopf algebra, playing the role of the ring of functions on the variety of solutions.

As an example, we consider here the Dyson–Schwinger equation (3.3), with  $B^+ = B_{\mathfrak{b}}^+$  the grafting operator to a vertex labelled with the bracketing operation. Assuming that parallel computations that do not feed into each other can be performed in arbitrary order, we see that this grafting operator descends to the commutative quotient  $\mathcal{H}_{\text{flow}, \mathcal{V}'}$  of the noncommutative Hopf algebra  $\mathcal{H}_{\text{flow}, \mathcal{V}'}^{nc}$ . Let  $X = \sum_{n \geq 1} x_n$  be the unique solution of (3.3), obtained as in (3.4), with  $x_1 = B_{\mathfrak{b}}^+(1) = v_{\mathfrak{b}}$  a single vertex labeled with the  $\mathfrak{b}$  operation. We define  $\mathcal{I}$  to be the ideal in the commutative algebra  $\mathcal{H}_{\text{flow}, \mathcal{V}'}$  generated by the  $x_n$  with  $n \geq 1$ .

The same conditions that ensures that  $\mathcal{A}$  is a Hopf subalgebra also ensure that  $\mathcal{I}$  is a Hopf ideal. We verify it in the case of (3.6). The more general case of [10] is similar.

**Lemma 3.2.** *The ideal  $\mathcal{I}$  generated by the components  $x_n$  with  $n \geq 1$  of the solution of (3.6) is a Hopf ideal.*

*Proof.* Elements of the ideal  $\mathcal{I}$  in  $\mathcal{H} = \mathcal{H}_{\text{flow}, \mathcal{V}'}$  are finite sums  $\sum_{m=1}^M h_m x_{k_m}$ , with  $h_m \in \mathcal{H}$  and  $x_k$  the coordinates of the unique solution of (3.6). The condition that  $\mathcal{I}$  is a Hopf ideal is that  $\Delta(\mathcal{I}) \subset \mathcal{I} \otimes \mathcal{H} \oplus \mathcal{H} \otimes \mathcal{I}$ . Using the formula (3.7) for the coproduct of the elements  $x_k$ , we see that we obtain a sum of terms of which the primitive part  $1 \otimes x_k + x_k \otimes 1$  is in  $\mathcal{H} \otimes \mathcal{I} \oplus \mathcal{I} \otimes \mathcal{H}$  and all the other terms  $\tilde{\Delta}(x_k)$  are in  $\mathcal{I} \otimes \mathcal{I}$ . Then the coproducts  $\Delta(h_m x_{k_m})$  will also be in  $\mathcal{H} \otimes \mathcal{I} \oplus \mathcal{I} \otimes \mathcal{H}$ .  $\square$

Notice that we can also work with the noncommutative Hopf algebra  $\mathcal{H}_{\text{flow}, \mathcal{V}'}^{nc}$  and the two sided ideal with elements of the form  $\sum_m h_m x_{k_m} \ell_m$ , with  $h_m, \ell_m \in \mathcal{H}_{\text{flow}, \mathcal{V}'}^{nc}$ , but for the considerations that follow (see §3.4 below) it is more natural to work with the commutative quotient.

**3.4. Yanofsky’s Galois theory of algorithms.** As we have seen above, one can pass to commutative quotients  $\mathcal{H}_{\text{flow}}$  of the noncommutative Hopf algebras of flow charts. The meaning of passing to the commutative quotient  $\mathcal{H}_{\text{flow}}$  is best expressed as follows: one can think of monomials in  $\mathcal{H}_{\text{flow}}^{nc}$  as diagrams for parallel computations of a certain number of outputs (one for each planar tree in the forest). Imposing commutativity then corresponds to the reasonable expectation that if one has two parallel computations that do not feed one into the other, then computing them in either order will not affect the result. However, when one introduces operations that graft the output of one three to the input of another, with  $\mathfrak{c}$  and  $\mathfrak{r}$  labels at the vertices, imposing the commutativity

relation is no longer appropriate, as the result of grafting would not be well defined as a *planar* tree. In the case of grafting with  $\mathfrak{c}$  label, we can still pass to the commutative quotient, as argued above.

**3.4.1. Commutative Hopf algebras and affine group schemes.** An advantage of working with commutative Hopf algebras is that they are dual to affine group schemes. Thus, the Hopf algebra  $\mathcal{H}_{\text{flow}}$  determines an affine group scheme  $G_{\text{flow}}$  such that, for any commutative  $\mathbb{K}$ -algebra  $A$ , one obtains a group  $G_{\text{flow}}(A) = \text{Hom}(\mathcal{H}_{\text{flow}}, A)$ , the set of homomorphisms of  $\mathbb{K}$ -algebras, with the product  $(\phi_1 \star \phi_2)(x) = \langle \phi_1 \otimes \phi_2, \Delta(x) \rangle$  dual to the coproduct and the inverse determined by the antipode.

**3.4.2. Galois groups of algorithms.** The idea elaborated by Yanofsky in the recent work [29], of a Galois theory of algorithms, aims at considering all possible “reasonably sets of relations” on the Hopf algebra  $\mathcal{H}_{\text{flow}}^{nc}$  of flow charts that would correspond to “implementing the same algorithm”, viewing “algorithms” as an intermediate level between the labelled planar rooted trees and the primitive recursive functions they compute.

Reasonable relations between the planar rooted trees of  $\mathcal{H}_{\text{flow}}^{nc}$  correspond therefore to passing to quotient Hopf algebras. Each such quotient is determined by a Hopf ideal  $\mathcal{I}$  in  $\mathcal{H}_{\text{flow}}^{nc}$ , the kernel of the map to the quotient Hopf algebra  $\mathcal{H}_{\mathcal{I}}$ .

In the commutative setting  $\mathcal{H}_{\text{flow}}$  one can view, dually, the quotient Hopf algebras  $\mathcal{H}_{\mathcal{I}}$  as subgroup schemes  $G_{\mathcal{I}}$  of the affine group scheme of characters  $G_{\text{flow}}$ . The group schemes  $G_{\mathcal{I}}$  obtained in this way are the basic objects of interest in Yanofsky’s Galois theory of algorithms.

**3.4.3. Hopf ideals and Galois groups.** In light of Yanofsky’s point of view, considering ideals in  $\mathcal{H}_{\text{flow}}$  generated by solutions of Dyson–Schwinger equations, and the resulting quotient Hopf algebra, rather than sub-Hopf algebras, corresponds to considering the “ring of function” of the variety of solutions. Namely the associated group scheme  $G_{\mathcal{I}}$  consists of those symmetries in  $G_{\text{flow}}$  that are compatible with the self-similarity structure described by the Dyson–Schwinger equation.

**Proposition 3.3.** *A Dyson–Schwinger equation (3.6) in the commutative Hopf algebra  $\mathcal{H}_{\text{flow}}$  determines a Galois group  $G_P \subset G_{\text{flow}}$ , in the sense of Yanofsky [29].*

The result follows directly from Lemma 3.2. A similar result holds for Dyson–Schwinger equations (3.3) with (3.8).

**3.5. A non-Hopf example with binary trees.** In the setting of Dyson–Schwinger equations described above, it is crucial that we consider the planar rooted trees in  $\mathcal{H}_{\text{flow}, \mathcal{V}}^{nc}$ , without external edges. In fact, this allows us to define the grafting operators  $B_{\delta}^{+}$  as grafting any number of trees to the *same* root consisting of a single vertex decorated by  $\delta$ . If we reintroduce external edges, then the graphs consisting of a single vertex but with different numbers of incoming edges are viewed as different generators in the Hopf algebra and they no longer define a cocycle  $B_{\delta}^{+}$ .

To see this more explicitly, we focus on the case of binary trees, keeping track of external edges. Consider the following primitive elements in the Hopf algebra  $\mathcal{H}_{\text{flow}, \mathcal{V}}^{nc, 0}$ :

(3.9) 

For each of these graphs  $\tau$  one can define a grafting operation  $B_\tau^+$  by setting the result equal to zero on monomials  $T$  that cannot be grafted to  $\tau$ . This  $B_\tau^+$  clearly does *not* satisfy the cocycle condition because one would have to have  $B_\tau^+(1) = 0$ , but then the expressions  $\tilde{\Delta}B_\tau^+$  and  $(id \otimes B_\tau^+)\tilde{\Delta}$ , when applied to elements  $T = \tau_1 \amalg \tau_2$  that are not in the kernel, would differ by a term corresponding to the cut that separates  $\tau$  from  $T$ . On the other hand, if  $B_\tau^+(1) = \tau$  as this case would require, then the cocycle condition fails on elements in the kernel of  $B_\tau^+$ .

Another way of defining a grafting operator  $B_\tau^+$  with  $\tau$  one of the trees (3.9) is to define, for a monomial  $T = \tau_1 \amalg \dots \amalg \tau_n$  the grafting  $B_\tau^+(T)$  to be obtained by just gluing one of the outputs of  $T$  (say, the first one) to one of the two inputs of  $\tau$  (say, the first one). We use this choice in the explicit example below. Again, however, one sees that the grafting defined in this way does *not* satisfy the cocycle condition. In fact, consider the case of a monomial  $T$  consisting of a forest with more than one component. Then only one of the components of  $T$  is glued to  $\tau$  by  $B_\tau^+$ . This means that, in the computation of the term  $\tilde{\Delta}B_\tau^+$  there is not just one but several admissible cuts that separate  $\tau$  from  $T$ , namely all the admissible cuts that include a cut of the edge used for grafting together with an arbitrary cut of the remaining components not connected to  $\tau$ . Thus, the difference between  $\tilde{\Delta}B_\tau^+$  and  $(id \otimes B_\tau^+)\tilde{\Delta}$  does not consist only of the term  $id \otimes B_\tau^+(1) = id \otimes \tau$ .

However, one can still consider a Dyson-Schwinger equation of the form

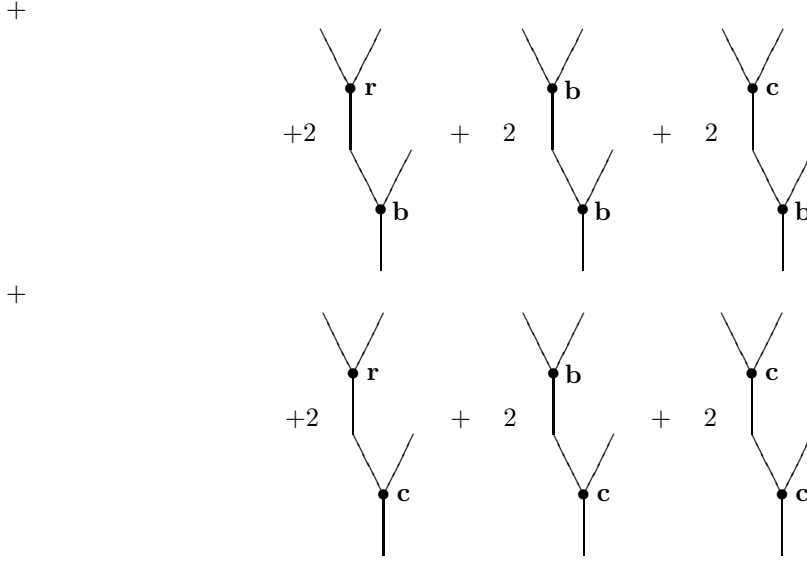
$$X = 1 + \sum_{\delta} B_{\tau_\delta}^+(X^2),$$

where the sum is on the three labels  $\delta \in \{\mathbf{b}, \mathbf{c}, \mathbf{r}\}$  of the vertex of the tree  $\tau_\delta$  as in (3.9). This admits an explicit solution, as in the general case above, which are of the form

$$x_{n+1} = \sum_{k=0}^n \sum_{\delta} B_{\tau_\delta}^+(x_k x_{n-k}).$$

In  $\overline{\mathcal{H}}_{\text{flow}, \nu}^0$ , one can see that the first few terms are of the form  $x_0 = I$  and

$$\begin{aligned}
 x_1 &= \begin{array}{c} \diagup \quad \diagdown \\ \bullet \\ | \\ \bullet \\ | \end{array} \mathbf{r} + \begin{array}{c} \diagup \quad \diagdown \\ \bullet \\ | \\ \bullet \\ | \end{array} \mathbf{b} + \begin{array}{c} \diagup \quad \diagdown \\ \bullet \\ | \\ \bullet \\ | \end{array} \mathbf{c} \\
 x_2 &= \sum_a B_+^a(2x_1 x_0) \\
 &= 2 \begin{array}{c} \diagup \quad \diagdown \\ \bullet \\ | \\ \diagup \quad \diagdown \\ \bullet \\ | \end{array} \mathbf{r} + 2 \begin{array}{c} \diagup \quad \diagdown \\ \bullet \\ | \\ \diagup \quad \diagdown \\ \bullet \\ | \end{array} \mathbf{b} + 2 \begin{array}{c} \diagup \quad \diagdown \\ \bullet \\ | \\ \diagup \quad \diagdown \\ \bullet \\ | \end{array} \mathbf{c}
 \end{aligned}$$



One can continue with the successive coefficients, such as  $x_3 = \sum_a B_+^a(x_1^2 + 2x_2x_0)$ , using the facts that for a vertex labeled by recursion, the order of inputs always matters, as well as for composition, while we can assume that order of inputs does not matter for bracketing as a consequence of the nature of parallel computation.

This example shows that, if we want to work with binary trees with external edges, which is a very natural choice from the point of view of computation theory, and we want to obtain a grafting operation that is non-trivial on monomials of arbitrary degree, we would have to extend the grafting to include, besides the primitive two-valent trees (3.9), also a choice of binary trees with larger numbers of input flags. This cannot be accommodated in the usual setting of combinatorial Dyson–Schwinger equations based on Hochschild cocycles in Hopf algebras. However, we shall see below that such generalizations exist naturally when one reformulates Dyson–Schwinger equations in the operadic setting.

#### 4. OPERADIC VIEWPOINT

Instead of using a Hopf algebraic setup to describe flow charts and primitive recursive functions, one can also adopt an operadic viewpoint, as suggested by Manin in [24].

Let us consider, as above, planar rooted trees  $\tau$  with vertex labeling by the elementary operations of type  $\{\mathbf{b}, \mathbf{c}, \mathbf{r}\}$ , oriented so that each tree has a single outgoing flag attached to the root vertex and a certain number of incoming flags.

**Definition 4.1.** *We define the operad of flow charts  $\mathcal{O}_{\text{flow}, \mathcal{V}}$  by setting  $\mathcal{O}(n)$  to be the  $\mathbb{K}$ -vector space spanned by labelled planar rooted trees with  $n$  incoming flags and the operad composition operations*

$$\circ_{\mathcal{O}} : \mathcal{O}(n) \otimes \mathcal{O}(m_1) \otimes \cdots \otimes \mathcal{O}(m_n) \rightarrow \mathcal{O}(m_1 + \cdots + m_n)$$

*are given on generators  $\tau \otimes \tau_1 \otimes \cdots \otimes \tau_n$  by grafting the output flag of the tree  $\tau_i$  to the  $i$ -th input flag of the tree  $\tau$ .*

As pointed out in [2], Dyson–Schwinger equations admit a natural operadic interpretation. Namely, given a formal series  $P(t) = 1 + \sum_{k=1}^{\infty} a_k t^k$  and a collection  $\beta = (\beta_n)$  with  $\beta_n \in \mathcal{O}(n)$ , we consider the equation

$$(4.1) \quad X = \beta(P(X)),$$

where  $X = \sum_k x_k$  is a formal sum with  $x_k \in \mathcal{O}(k)$ . In the right hand side we have  $\beta(P(X))_1 = 1 + \beta_1 \circ x_1$ , where 1 is the identity in  $\mathcal{O}(1)$ , and for  $n \geq 2$

$$(4.2) \quad \beta(P(X))_n = \sum_{k=1}^n \sum_{j_1 + \dots + j_k = n} a_k \beta_k \circ (x_{j_1} \otimes \dots \otimes x_{j_k}),$$

with  $x_{j_1} \otimes \dots \otimes x_{j_k} \in \mathcal{O}(j_1) \otimes \dots \otimes \mathcal{O}(j_k)$ , so that the composition  $\beta_k \circ_{\mathcal{O}} (x_{j_1} \otimes \dots \otimes x_{j_k}) \in \mathcal{O}(n)$ , since  $j_1 + \dots + j_k = n$ .

**Proposition 4.2.** *For  $\mathcal{O} = \mathcal{O}_{\text{flow}, \mathcal{V}}$ , the operad of flow charts, if  $a_1 \beta_1 \neq 1 \in \mathcal{O}(1)$ , the operadic Dyson–Schwinger equation (4.1) has a unique solution  $X \in \prod_{n \geq 1} \mathcal{O}(n)$  given inductively by*

$$(4.3) \quad (1 - a_1 \beta_1) \circ x_{n+1} = \sum_{k=2}^{n+1} \sum_{j_1 + \dots + j_k = n+1} a_k \beta_k \circ (x_{j_1} \otimes \dots \otimes x_{j_k}).$$

*Proof.* First observe that in the operad of flow charts  $\mathcal{O}(1)$  is one-dimensional, since it is spanned by the trees with a single vertex, one incoming and one outgoing flag, with the operations  $\mathbf{b}$ ,  $\mathbf{c}$  or  $\mathbf{r}$  at the vertex, but in the case of a single input, all of these operations are the identity, so that elements of  $\mathcal{O}(1)$  are scalar multiples of the identity map, viewed as operations on primitive recursive functions. Thus, the element  $\beta_1 \in \mathcal{O}(1)$  is a scalar multiple of the identity and, if we assume that  $a_1 \beta_1 \neq 1$  in  $\mathcal{O}(1)$ , then  $1 - a_1 \beta_1$  is invertible. Since we have  $\beta(P(X))_1 = 1 + \beta_1 \circ x_1$  as the term in  $\mathcal{O}(1)$  in the right hand side of (4.1), we obtain  $(1 - a_1 \beta_1)x_1 = 1$ , which fixes  $x_1 = (1 - a_1 \beta_1)^{-1} \in \mathcal{O}(1)$ . At the next step we have, from (4.2),  $x_2 = a_1 \beta_1 \circ x_2 + a_2 \beta_2 \circ (x_1 \otimes x_1)$ , which gives  $x_2 = (1 - a_1 \beta_1)^{-1} (a_2 \beta_2 \circ (x_1 \otimes x_1))$ , with  $x_1$  as above. At the  $(n+1)$ -st step,  $x_{n+1}$  is then determined uniquely by (4.3) in terms of the coefficients  $a_k$  and the elements  $\beta_k$  for  $k = 1, \dots, n+1$ .  $\square$

Let then  $\mathcal{O}_{\beta, P}(n)$  denote the  $\mathbb{K}$ -linear span of all the compositions  $x_k \circ (x_{j_1} \otimes \dots \otimes x_{j_k})$ , for  $k = 1, \dots, n$  and  $j_1 + \dots + j_k = n$ , with  $x_k$  the coordinates of the solution  $X = \sum_k x_k$  of the Dyson–Schwinger equation determined by  $\beta = (\beta_n)$  and  $P(t)$ . The  $\mathcal{O}_{\beta, P}(n)$  form a sub-operad under the composition maps induced from  $\mathcal{O}(n)$ .

For instance, for a choice of one of the operations  $\delta \in \{\mathbf{b}, \mathbf{c}, \mathbf{r}\}$ , we can take  $a_1 \neq 1$  and the element  $\beta_k$  given by the tree with a single vertex marked by  $\delta$ , with  $k$  incoming flags and one outgoing flag. This choice gives an operadic reformulation of the Hopf-theoretic Dyson–Schwinger equations with the cocycles  $B_{\delta}^+$ . However, we can now consider also more general operadic Dyson–Schwinger equations for different choices of  $\beta = (\beta_n)$  that do not correspond to Hochschild cocycles in the Hopf algebra setting.

Restricting to the sub-operad  $\mathcal{O}_{\beta, P}(n)$  generated by solutions of the Dyson–Schwinger equation should be regarded as considering those operations (formal combinations of flow charts) that satisfy a self-similarity property with respect to the transformation  $X \mapsto \beta(P(X))$ .

4.0.1. *The case of binary trees.* In particular, we can now revisit the example of the binary trees in this operadic setting. Let  $\mathcal{O}_{\text{flow}, \mathcal{V}}^0(n)$  be the  $\mathbb{K}$ -linear space spanned by the planar *binary* trees with vertex labeling by the binary operations  $\mathbf{b} = \mathbf{b}_2$ ,  $\mathbf{r}$ ,  $\mathbf{c}$ . These form a sub-operad of  $\mathcal{O}_{\text{flow}, \mathcal{V}'}$  with the induced composition maps.

For a fixed  $\delta \in \{\mathbf{b}, \mathbf{r}, \mathbf{c}\}$  let  $\beta_{1, \delta}$  be the binary tree with a single vertex, two incoming and one outgoing flag, as in (3.9). We then consider a choice  $\beta_n$  of a binary tree with  $n$  input flags, for each  $n \geq 1$ , with a fixed choice of vertex labelings in  $\{\mathbf{b}, \mathbf{r}, \mathbf{c}\}$ , not necessarily all of the same type. Each such choice of  $\beta = (\beta_n)$ , together with a choice of  $P(t)$ , now determines an operadic Dyson–Schwinger equation of the form (4.1), in the binary setting, with a corresponding solution as in Proposition 4.2.

4.0.2. *Systems of Dyson–Schwinger equations in operads.* By analogy to what happens in Hopf algebras, [11], one can pass from the case of a single Dyson–Schwinger equation to systems of Dyson–Schwinger equations. In the operadic setting a system of Dyson–Schwinger equations would be determined by the data of formal series  $F_\delta(t_1, t_2, t_3) = \sum_{k_1, k_2, k_3} a_{k_1, k_2, k_3}^{(\delta)} t_1^{k_1} t_2^{k_2} t_3^{k_3}$  and of elements  $\beta^{(\delta)} = (\beta_n^{(\delta)})$ , where for  $\delta \in \{\mathbf{b}, \mathbf{c}, \mathbf{r}\}$ , the elements  $\beta_n^{(\delta)} \in \mathcal{O}(n)$  are realized by rooted trees with root vertex marked by  $\delta$ . The operadic systems of Dyson–Schwinger equations are then of the form

$$X_\delta = \beta^{(\delta)}(F_\delta(X)).$$

The explicit form of the solutions is more cumbersome than in the case of a single equation in Proposition 4.2 above, and it follows a pattern similar to the derivation of the explicit solutions of Hopf theoretic systems of Dyson–Schwinger equations (3.5) as in [11]. These systems detect more elaborate forms of self-similarity in the operad of flow charts, involving different families of grafting operations simultaneously.

4.0.3. *Operads and Properads.* As mentioned in [24], one can extend the Hopf algebra of flow charts by considering, instead of labelled embedded rooted trees, more general labelled embedded acyclic graphs, namely graphs endowed with an acyclic orientation. Correspondingly, the operad of flow charts would be replaced by a properad, where the compositions extend from grafting output and input flags of trees to grafting outputs and inputs of acyclic graphs.

The notion of *properad* was introduced in [27] as an intermediate notion between operad and prop. Namely, a properad parameterizes operations with varying numbers of inputs and outputs that are labelled by connected acyclic graphs, while the operad case uses trees (varying number of inputs and a single output) and props allow for disconnected graphs.

More precisely, we consider the properad  $\mathcal{P}_{\text{flow}, \mathcal{V}'}$  where  $\mathcal{P}(m, n)$  is the  $\mathbb{K}$ -vector space spanned by planar connected directed (acyclic) graphs with  $m$  incoming flags and  $n$  outgoing flags, with vertices decorated by operation that include the elementary  $\mathbf{b}$ ,  $\mathbf{c}$ ,  $\mathbf{r}$ , with  $m$  inputs and one output, but also include now a chosen set of additional operations with  $m$  inputs and  $n$  outputs. As we have seen in the case of extending the operations  $\mathbf{c}$  and  $\mathbf{r}$  from binary to  $m$ -ary inputs, in this case also such additional operations with  $m$  inputs and  $n$  outputs acting on primitive recursive functions can be decomposed in terms of the elementary operations: the resulting operations associated to vertices with  $m$  incoming and  $n$  outgoing flags can be regarded as “macros”, in the sense discussed in §2 of [29]. The properad has composition operations

$$\mathcal{P}(m, n) \otimes \mathcal{P}(j_1, k_1) \otimes \cdots \otimes \mathcal{P}(j_\ell, k_\ell) \rightarrow \mathcal{P}(j_1 + \cdots + j_\ell, n), \quad \text{for } k_1 + \cdots + k_\ell = m.$$

4.0.4. *Dyson-Schwinger equations in properads.* We can define Dyson-Schwinger equations for properads, extending the operadic case considered above. Namely we consider again a choice of a formal power series  $P(t) = 1 + \sum_k a_k t^k$  and a collection  $\beta = (\beta_{m,n})$  of elements with  $\beta_{m,n} \in \mathcal{P}(m,n)$ . We then consider the equation

$$(4.4) \quad X = \beta(P(X))$$

as in (4.1), where now the right hand side has  $(m,n)$ -component in  $\mathcal{P}(m,n)$  given by

$$(4.5) \quad \beta(P(X))_{m,n} = \sum_{k=1}^m a_k \sum_{\substack{j_1 + \dots + j_k = m \\ i_1 + \dots + i_k = n}} \beta_{k,n} \circ (x_{j_1, i_1} \otimes \dots \otimes x_{j_k, i_k}).$$

In order to construct solutions of the properad Dyson-Schwinger equations, we introduce transformations  $\Lambda_n = \Lambda_n(a, \beta)$  with

$$\Lambda_n(a, \beta) : \oplus_{k=1}^n \mathcal{P}(n, k) \rightarrow \oplus_{k=1}^n \mathcal{P}(n, k), \quad \text{with} \quad \Lambda_n(a, \beta)_{ij} = a_j \beta_{j,i}.$$

**Theorem 4.3.** *If for all  $n \geq 1$  the transformation  $I - \Lambda_n(a, \beta)$  is invertible, with  $I$  the identity on  $\oplus_{k=1}^n \mathcal{P}(n, k)$ , then the properad Dyson-Schwinger equation (4.4) admits a unique solution, given by  $x_{1,1} = \Lambda_1^{-1}$ , and for  $m < n$  by*

$$(4.6) \quad x_{m,n} = \sum_{k=1}^m a_k \beta_{k,n} \circ \left( \sum_{\ell=1}^k \sum_{\substack{j_1 + \dots + j_\ell = m \\ i_1 + \dots + i_\ell = k}} x_{j_1, i_1} \otimes \dots \otimes x_{j_\ell, i_\ell} \right),$$

while the remaining components  $x_{m,n}$  with  $m \geq n$  are determined by

$$(4.7) \quad Y_n(x) = (I - \Lambda_n)^{-1} \Lambda_n V^{(n)}(x),$$

where  $Y_n(x)^t = (x_{n,1}, \dots, x_{n,n})$  and  $V^{(n)}(x)^t = (V^{(n)}(x)_j)_{j=1, \dots, n}$

$$V^{(n)}(x)_j = \sum_{k=2}^n \sum_{\substack{r_1 + \dots + r_k = n \\ s_1 + \dots + s_k = j}} x_{r_1, s_1} \otimes \dots \otimes x_{r_k, s_k}.$$

*Proof.* By (4.5) we see that we have  $x_{1,1} = 1 + a_1 \beta_{1,1} \circ x_{1,1}$ , which can be solved for  $x_{1,1}$  provided the invertibility of  $1 - \Lambda_1 = 1 - a_1 \beta_{1,1}$  holds. The components  $x_{m,n}$  with  $m < n$  are determined by (4.6), where the right hand side only involves components  $x_{r,s}$  with  $r < m$  and  $s < n$ . The equations for the remaining entries  $x_{m,n}$  with  $n \leq m$  break into systems of the form

$$(I - \Lambda_m) \begin{pmatrix} x_{m,1} \\ \vdots \\ x_{m,m} \end{pmatrix} = \sum_j (\Lambda_m)_{ij} \begin{pmatrix} \sum_{k=2}^m \sum_{\substack{r_1 + \dots + r_k = m \\ s_1 + \dots + s_k = j}} x_{r_1, s_1} \otimes \dots \otimes x_{r_k, s_k} \end{pmatrix}_j.$$

Provided that the invertibility condition for the transformation  $I - \Lambda_m$  holds, these systems provide unique values for all the components  $x_{m,n}$  with  $n \leq m$ , as a function of the components  $x_{r,s}$  with  $r < m$ , that we have already been determined.  $\square$

Notice that one does not expect the invertibility of  $I - \Lambda_n(a, \beta)$  to hold in general, as  $\Lambda_n(a, \beta)$  involves compositions with the elements  $\beta_{j,i}$ . The simplest example in which one has invertibility can be obtained by taking  $\beta_{m,n} = 0$  for  $m \neq n$  and  $I - \beta_{n,n}$  an invertible transformation in  $\mathcal{P}(n, n)$ .

As in the operad case, we can define  $\mathcal{P}_{\beta,P}(m,n)$  to be the  $\mathbb{K}$ -vector spaced spanned by all the properad compositions

$$x_{k,n} \circ (x_{j_1,i_1} \otimes \cdots \otimes x_{j_\ell,i_\ell})$$

with  $j_1 + \cdots + j_\ell = m$  and  $i_1 + \cdots + i_\ell = k$  and where all the  $x_{m,n}$  are components of the solution to the properad Dyson–Schwinger equation determined by the choice of  $P(t)$  and of  $\beta = (\beta_{m,n})$ , with the invertibility condition of Theorem 4.3. The  $\mathcal{P}_{\beta,P}(m,n)$  form a sub-properad of  $\mathcal{P}_{\text{flow},\mathcal{V}'}$  with the induced composition.

## 5. RENORMALIZATION OF THE HALTING PROBLEM

In [22], Manin adapted the formalism of perturbative renormalization to the halting problem in the theory of computation. To summarize Manin’s approach, the basic idea is that one considers non-computable functions as an analog of divergent Feynman integrals. Renormalization consists of a procedure of extraction of finite values from divergences and in this setting it corresponds to extracting a “computable part” from a non-computable function.

**5.1. The halting problem: regularization and renormalization.** We describe here a setting for renormalization of the halting problem which is minor elaboration on the procedure described in §3 of [22].

**5.1.1. Partial recursive functions and Hopf algebras.** We first pass from the class of primitive recursive functions considered above to the larger class of partial recursive functions. These have a similar presentation as the one recalled in §2 above, with the same basic functions (successor, constants, and projections) and, in addition to the three elementary operations  $\mathbf{c}$ ,  $\mathbf{b}$ ,  $\mathbf{r}$  of composition, bracketing and recursion, an additional  $\mu$  operation that assigns to an input function  $f : \mathbb{N}^{n+1} \rightarrow \mathbb{N}$  assigns an output

$$h : \mathbb{N}^n \rightarrow \mathbb{N}, \quad h(x_1, \dots, x_n) = \min\{x_{n+1} \mid f(x_1, \dots, x_{n+1}) = 1\},$$

with domain

$$\mathcal{D}(h) = \{(x_1, \dots, x_n) \mid \exists x_{n+1} \geq 1 : f(x_1, \dots, x_{n+1}) = 1, \text{ with } (x_1, \dots, x_n, k) \in \mathcal{D}(f), \forall k \leq x_{n+1}\}.$$

According to Church’s thesis the partial recursive functions obtained in this way are exactly the semi-computable functions, namely those for which there exists a program that, for all  $x \in \mathcal{D}(f)$ , computes  $f(x)$  and which either computes zero or runs for an infinite time when  $x \notin \mathcal{D}(f)$ , see §V of [20]. Notice that it is the presence of the additional operation  $\mu$  that allows for the construction, starting from the basic functions, of functions that are only partially defined and semi-computable.

Correspondingly, we enrich the Hopf algebras of flow charts by additional vertex decorations by the  $\mu$  operations, extended to vertices of valence greater than two by a combination with the bracketing operation. We shall still use the notations  $\mathcal{H}_{\text{flow},\mathcal{R}'}^{nc}$  and  $\mathcal{H}_{\text{flow},\mathcal{V}'}^{nc}$ , for the resulting Hopf algebras with admissible decorations by partial recursive functions or with vertex decorations, respectively, and  $\mathcal{H}_{\text{flow},\mathcal{R}'}$  and  $\mathcal{H}_{\text{flow},\mathcal{V}'}$  for their commutative quotients.



5.1.2. *Algebraic renormalization.* The basic algebraic formalism for perturbative renormalization, in its formulation as given in [9] (see also §5 of [25]), consists of an *algebraic Feynman rule*

$$(5.1) \quad \phi : \mathcal{H} \rightarrow \mathcal{B}$$

which is a *morphism of commutative algebras* from a *commutative Hopf algebra*  $\mathcal{H}$  to a *Rota–Baxter algebra*  $\mathcal{B}$  of weight  $-1$ .

Recall that a commutative Rota–Baxter algebra of weight  $\lambda$  is a commutative, associative algebra  $\mathcal{B}$  endowed with a linear operator  $T$  on it that satisfies the identity

$$(5.2) \quad T(x)T(y) = T(xT(y)) + T(T(x)y) + \lambda T(xy).$$

The field of convergent Laurent series in one variable with the projection  $T$  onto the polar part is an example of Rota–Baxter algebra of weight  $\lambda = -1$ , which is widely used in renormalization theory, [3], [5]. The Rota–Baxter operator  $T$  of weight  $-1$  determines a splitting of  $\mathcal{B}$  into two commutative unital subalgebras,  $\mathcal{B}_+ = (1 - T)\mathcal{B}$  and  $\mathcal{B}_-$  given by  $T\mathcal{B}$  with a unit adjoined. The fact that  $\mathcal{B}_\pm$  are indeed algebras and not just vector spaces follows from the Rota–Baxter identity (5.2).

The BPHZ renormalization of the algebraic Feynman rule (5.1) is then obtained by showing the existence of a multiplicative factorization

$$\phi = (\phi_- \circ S) \star \phi_+, \quad \text{with} \quad \phi_\pm : \mathcal{H} \rightarrow \mathcal{B}_\pm,$$

obtained inductively via the BPHZ preparation formula (see [3], [9])

$$(5.3) \quad \phi_-(x) = -T(\phi(x) + \sum \phi_-(x')\phi(x'')), \quad \text{and} \quad \phi_+(x) = (1 - T)(\phi(x) + \sum \phi_-(x')\phi(x'')),$$

where  $x \in \mathcal{H}$ , with  $\Delta(x) = 1 \otimes x + x \otimes 1 + \sum x' \otimes x''$ . The factorization is unique if normalized by  $\epsilon_- \circ \phi_- = \epsilon$ , where  $\epsilon_- : \mathcal{B}_- \rightarrow \mathbb{C}$  is the augmentation map and  $\epsilon$  is the counit of  $\mathcal{H}$ . Again, it is the Rota–Baxter property of  $(\mathcal{B}, T)$  that ensures that  $\phi_\pm$  obtained in this way are algebra homomorphisms. The homomorphism  $\phi_+ : \mathcal{H} \rightarrow \mathcal{B}_+$  is the *renormalized algebraic Feynman rule*, while  $\phi_- : \mathcal{H} \rightarrow \mathcal{B}_-$  is the *counterterm*, the divergence.

5.1.3. *An algebraic Feynman rule for the halting problem.* We let  $\mathcal{B}$  be the algebra of functions  $\Phi : \mathbb{N}^m \rightarrow \mathcal{M}(D)$  from  $\mathbb{N}^m$ , for some  $k$ , to the algebra  $\mathcal{M}(D)$  of analytic functions in the unit disk  $D = \{z \in \mathbb{C} : |z| < 1\}$ . Since the functions  $\Phi$  can have different domains  $\mathbb{N}^k$ , the sums and products are defined by extending them to functions  $\Phi(k_1, k_2, \dots, k_n, \dots)$  from a common domain  $\mathbb{N}^\infty$ , which depend on only finitely many variables,  $\Phi = \Phi \circ \pi_m^\infty$ , with  $\pi_m^\infty : \mathbb{N}^\infty \rightarrow \mathbb{N}^m$ . The Rota–Baxter operator  $T$  on  $\mathcal{B}$  is given by componentwise (for each  $k$ ) projection onto the polar part at  $z = 1$ . Thus,  $\mathcal{B}_+$  consists of functions from  $\mathbb{N}^\infty$  (depending only on finitely many  $k_i$ ) to  $\mathcal{M}(D)_+$ , the algebra of meromorphic functions in  $D$  that extend continuously at  $z = 1$ , while  $\mathcal{B}_-$  consists of functions that are either constant or have a pole at  $z = 1$ .

We define algebraic Feynman rules for the Hopf algebras  $\mathcal{H}_{\text{flow}, \mathcal{R}'}$  and  $\mathcal{H}_{\text{flow}, \mathcal{V}'}$ . In the case of  $\mathcal{H}_{\text{flow}, \mathcal{R}'}$ , where both vertices and flags in the trees are labeled by admissible labelings, there is a well defined partial recursive function  $f$  associated to each labelled rooted tree  $\tau$  in  $\mathcal{H}_{\text{flow}, \mathcal{R}'}$ , namely the function  $f$  labeling the unique outgoing flag of  $\tau$ . Let us first assume that the partial recursive function  $f$  has range in  $\mathbb{N}$ . We extend this  $f : \mathbb{N}^m \rightarrow \mathbb{N}$  to a function  $\bar{f} : \mathbb{N}^m \rightarrow \mathbb{Z}_{\geq 0}$  that computes  $f(x)$  at  $x \in \mathcal{D}(f)$  and takes value 0 at  $x \notin \mathcal{D}(f)$ .

We assign to a tree  $\tau$  that computes  $f$  an element in  $\Phi_\tau(\underline{k}, z)$  in  $\mathcal{B}$ , defined (as in §3 of [22]) by

$$(5.4) \quad \Phi_\tau(\underline{k}, z) = \Phi(\underline{k}, f, z) := \sum_{n \geq 0} \frac{z^n}{(1 + n f(\underline{k}))^2}.$$

The resulting function  $\Phi_\tau(\underline{k}, \cdot) \in \mathcal{M}(D)$  has a pole at  $z = 1$  iff  $\underline{k} \notin \mathcal{D}(f)$ . We extend this definition multiplicatively to cases where the output of the flow chart  $\tau$  is a partial recursive function  $f : \mathbb{N}^k \rightarrow \mathbb{N}^\ell$ , by setting  $\Phi_\tau(\underline{k}, z) = \prod_{j=1}^\ell \Phi(\underline{k}, f_j, z)$ . In this way, the resulting function has a pole at  $z = 1$  if at least one of the  $f_j$  does, that is, iff  $\underline{k} \notin \mathcal{D}(f_1) \cap \dots \cap \mathcal{D}(f_\ell) = \mathcal{D}(f)$ , consistently with the above. We then extend the definition multiplicatively to the case of a monomial  $\tau_1 \cdots \tau_n$  in  $\mathcal{H}_{\text{flow}, \mathcal{R}'}$  and additively to linear combinations of monomials. Thus, we obtain an algebra homomorphism  $\Phi : \mathcal{H}_{\text{flow}, \mathcal{R}'} \rightarrow \mathcal{B}$ .

We write  $\Phi(\underline{k}, f, z) := \Phi_f(\underline{k})(z)$  for the function  $\Phi_f(\underline{k}) \in \mathcal{M}(D)$  of  $z \in D$ , associated to a given  $\underline{k} \in \mathbb{N}^\infty$  and depending on the output  $f$  of a given flow chart, with  $f$  a partial recursive functions. If we denote by  $\tau$  the decorated planar rooted tree describing the flowchart, we equivalently write  $\Phi(\underline{k}, \tau, z)$  or  $\Phi_{\underline{k}, \tau}(z)$ .

The renormalization of the halting problem described by Manin in [22] then consists of applying the BPHZ factorization (5.3) to these algebraic Feynman rules to consistently extract a computable part from the halting problem for partial recursive functions that accounts for “subdivergences” created by subroutines within the flowcharts.

When we apply the BPHZ procedure to  $\Phi$  defined as in (5.4), we obtain

$$(5.5) \quad \Phi_-(\underline{k}, f_\tau, z) = -T(\Phi(\underline{k}, f_\tau, z) + \sum_C \Phi_-(\underline{k}, f_{\pi_C(\tau)}, z) \Phi(\underline{k}, f_{\rho_C(\tau)}, z)).$$

Because  $f = f_\tau$  is the output function computed by the flow chart, that is, the label of the outgoing flag of  $\tau$ , we have  $f_{\rho_C(\tau)} = f_\tau$ , since these trees have the same root and outgoing flag with the same label. Thus, the expression above is simply of the form

$$(5.6) \quad \Phi_-(\underline{k}, f_\tau, z) = -T \left( \Phi(\underline{k}, f_\tau, z) \left( 1 + \Phi_-(\underline{k}, \sum_C f_{\pi_C(\tau)}, z) \right) \right).$$

This has the effect of considering not only the pole at  $z = 1$  that is caused by  $f_\tau$  itself, but also those that come from subdivergences created by all the partial recursive functions  $f_{\pi_C(\tau)}$  computed in the intermediate steps of the computation performed by the flow chart  $\tau$ , which are the outputs of the “pruned parts”  $\pi_C(\tau)$  of the admissible cuts on  $\tau$ .

**5.1.4. The case of  $\mathcal{H}_{\text{flow}, \mathcal{V}'}$ .** A variation on the construction above would be to define an algebraic Feynman rule on  $\mathcal{H}_{\text{flow}, \mathcal{V}'}$ , where the flags are not labelled and only vertices are. In this case, in order to define something like (5.4), we need a way of assigning inputs to the flow charts. A reasonable choice is to use the basic functions as input. We then define the algebraic Feynman rule as

$$(5.7) \quad \Phi(\underline{k}, \tau, z) := \prod_{\sigma} \Phi(\underline{k}, f_{\tau, \sigma}, z),$$

where the product is over  $\sigma$  ranging over the set of all possible functions  $\sigma : E_{\text{ext}}^{\text{in}}(\tau) \rightarrow \{s, c, \pi\}$  that label the incoming external edges of  $\tau$  by basic functions (successor, constant, or projection). The partial recursive function  $f_{\tau, \sigma}$  is the output of the flow chart given by the tree  $\tau$  with inputs assigned by  $\sigma$ . Each  $\Phi(\underline{k}, f_{\tau, \sigma}, z)$  is computed as in (5.4), and (5.7) is extended to arbitrary elements

of the Hopf algebra as before. In this setting, the function  $\Phi_\tau(\underline{k}, z)$  has a pole at  $z = 1$  iff there is at least a choice of a basic input  $\sigma$  for which  $\underline{k} \notin \mathcal{D}(f_{\tau,\sigma})$ .

With this setting, the BPHZ formula for the algebraic Feynman rule (5.7) becomes more interesting than in the case of (5.6). We have again (5.5), in the form

$$(5.8) \quad \Phi_-(\underline{k}, \tau, z) = -T(\Phi(\underline{k}, \tau, z) + \sum_C \Phi_-(\underline{k}, \pi_C(\tau), z) \Phi(\underline{k}, \rho_C(\tau), z)),$$

with  $\Phi(\underline{k}, \rho_C(\tau), z)$  and  $\Phi(\underline{k}, \pi_C(\tau), z)$  again computed as in (5.7), but this time it is no longer true that  $f_\tau = f_{\rho_C(\tau)}$  because on the tree  $\rho_C(\tau)$  we are using the new inputs given by basic functions and not the input coming from the output of  $\pi_C(\tau)$ . The divergence (5.8) here combines the divergences of the  $f_{\tau,\sigma}$  with divergences coming from partial recursive functions  $f_{\pi_C(\tau),\sigma}$  and  $f_{\rho_C(\tau),\sigma}$  that flowcharts  $\pi_C(\tau)$  and  $\rho_C(\tau)$  compute starting from inputs of basic functions.

**5.2. Dyson–Schwinger equations for the halting problem.** If we work with the Feynman rule described above on the Hopf algebra  $\mathcal{H}_{\text{flow}, \mathcal{V}'}$  with only vertex decorations, then we can readily consider Dyson–Schwinger equations as discussed in §3 above. However, if we work with the definition of the algebraic Feynman rule  $\Phi$  described in (5.4) on the Hopf algebra  $\mathcal{H}_{\text{flow}, \mathcal{R}'}$  of flow charts with flag decorations, we need to extend the grafting operators appropriately that are needed to define Dyson–Schwinger equations from  $\mathcal{H}_{\text{flow}, \mathcal{V}'}$  to  $\mathcal{H}_{\text{flow}, \mathcal{R}'}$ . This can be done by assigning as output the empty function  $f = \emptyset$ , mapped to the constant function  $\Phi(\underline{k}, \emptyset, z) \equiv 1$ , as the output of any tree  $\tau$  that is obtained through a grafting where the flag labels don’t match.

We can then consider a Dyson–Schwinger equation of the form (3.3), with  $B^+ = \sum_\delta B_\delta^+$ , with  $\delta \in \{\mathfrak{b}, \mathfrak{c}, \mathfrak{r}, \mu\}$  or a more general system of Dyson–Schwinger equations (3.5), with the condition that the components  $x_m$  of the unique solution span a Hopf subalgebra of  $\mathcal{H}_{\text{flow}, \mathcal{R}'}$ . In such case, we can restrict the algebraic Feynman rule  $\Phi$  constructed above to this Hopf subalgebra and still perform the BPHZ renormalization. The divergences  $\Phi_-(\underline{k}, f_{x_m}, z)$  (or  $\Phi_-(\underline{k}, f_{x_m,\sigma}, z)$  in the case of  $\mathcal{H}_{\text{flow}, \mathcal{V}'}$ ) measure the “amount of non-computability” that can be produced by flow charts that satisfy the self-similarity property with respect to the operations  $\{\mathfrak{b}, \mathfrak{c}, \mathfrak{r}, \mu\}$  specified by the Dyson–Schwinger equation.

**Acknowledgment.** The first author was supported for this project by the Summer Undergraduate Research Fellowship (SURF) program of Caltech, through a Herbert J. Ryser fellowship. The second author is partially supported by NSF grants DMS-0901221, DMS-1007207, DMS-1201512, and PHY-1205440. The second author acknowledges MSRI for hospitality and support.

## REFERENCES

- [1] J. Baez and M. Stay, *Physics, Topology, Logic and Computation: A Rosetta Stone*, in “New structures for physics”, 95–172, Lecture Notes in Phys., 813, Springer, 2011.
- [2] C. Bergbauer and D. Kreimer, *Hopf algebras in renormalization theory: locality and Dyson-Schwinger equations from Hochschild cohomology*, in “Physics and Number Theory”, 133–164, IRMA Lect. Math. Theor. Phys., 10, Eur. Math. Soc., 2006.
- [3] A. Connes, D. Kreimer, *Renormalization in quantum field theory and the Riemann-Hilbert problem. I. The Hopf algebra structure of graphs and the main theorem*. Comm. Math. Phys. 210 (2000), no. 1, 249–273.
- [4] A. Connes, D. Kreimer, *Insertion and elimination: the doubly infinite Lie algebra of Feynman graphs*. Ann. Henri Poincaré 3 (2002), no. 3, 411–433.
- [5] A. Connes, M. Marcolli, *Noncommutative Geometry, Quantum Fields and Motives*, Colloquium Publications, Vol.55, American Math. Soc., 2008.

- [6] A. Connes, M. Marcolli, *Renormalization and motivic Galois theory*, Int. Math. Res. Not. 2004, no. 76, 4073–4091.
- [7] F. Dyson, *The S-matrix in quantum electrodynamics*, Phys. Rev. 75 (1949) 1736–1755.
- [8] K. Ebrahimi-Fard, D. Kreimer, I. Mencattini, *On the insertion-elimination Lie algebra of Feynman graphs*, in “Lie theory and its applications in physics V”, 124–134, World Sci., 2004.
- [9] K. Ebrahimi-Fard, L. Guo, D. Kreimer, *Integrable renormalization. II. The general case*. Ann. Henri Poincaré 6 (2005), no. 2, 369–395.
- [10] L. Foissy, *Faà di Bruno subalgebras of the Hopf algebra of planar trees from combinatorial Dyson–Schwinger equations*, Advances in Math. 218 (2008) 136–162.
- [11] L. Foissy, *Classification of systems of Dyson–Schwinger equations in the Hopf algebra of decorated rooted trees*, Advances in Math. 224 (2010) 2094–2150.
- [12] L. Foissy, *Lie algebras associated to systems of Dyson–Schwinger equations*, Advances in Math. 226 (2011) 4702–4730.
- [13] L. Foissy, *Les algèbres de Hopf des arbres enracinés, I*, Bull. Sci. Math. 126 (2002) 193–239.
- [14] R. Holtkamp, *Comparison of Hopf algebras on trees*, Arch. Math. (Basel) 80 (4) (2003) 368–383.
- [15] D. Kreimer, *On the Hopf algebra structure of perturbative quantum field theories*, Adv. Theor. Math. Phys. 2 (1998), no. 2, 303–334.
- [16] D. Kreimer, *The core Hopf algebra*, in “Quanta of maths”, 313–321, Clay Math. Proc., 11, Amer. Math. Soc., 2010.
- [17] D. Kreimer, *Dyson–Schwinger equations: from Hopf algebras to number theory*, in “Universality and renormalization”, 225–248, Fields Inst. Commun., 50, Amer. Math. Soc., 2007.
- [18] D. Kreimer, W. van Suijlekom, *Recursive relations in the core Hopf algebra*, Nuclear Phys. B 820 (2009), no. 3, 682–693.
- [19] D. Kreimer, K. Yeats, *An étude in non-linear Dyson–Schwinger equations*, Nuclear Phys. B Proc. Suppl. 160 (2006), 116–121.
- [20] Yu.I. Manin, *A Course in Mathematical Logic for Mathematicians*, Graduate Texts in Mathematics, Second Edition, 2010.
- [21] Yu.I. Manin, *Renormalization and computation I: Motivation and background*, arXiv:0904.4921.
- [22] Yu.I. Manin, *Renormalization and computation II: Time cutoff and the halting problem*, arXiv:0908.3430.
- [23] Yu.I. Manin, *Infinites in quantum field theory and in classical computing: renormalization program*, preprint, 2010.
- [24] Yu.I. Manin, *Zipf’s law and Levin’s probability distributions*, arXiv:1301.0427.
- [25] M. Marcolli, *Feynman motives*, World Scientific, 2010.
- [26] J. Schwinger, *On Green’s functions of quantized fields I, II*, PNAS 37 (1951) 452–459.
- [27] B. Vallette, *A Koszul duality for props*, Trans. Amer. Math. Soc. 359 (2007), no. 10, 4865–4943.
- [28] N. Yanofsky, *Towards a definition of an algorithm*, J. Logic Comput. 21 (2011), no. 2, 253–286.
- [29] N. Yanofsky, *Galois theory of algorithms*, arXiv:1011.0014.
- [30] K. Yeats, *Rearranging Dyson–Schwinger Equations*, Memoirs of the American Mathematical Society, Vol.211, American Mathematical Society, 2011.

PHYSICS DEPARTMENT, CALTECH, 1200 E. CALIFORNIA BLVD. PASADENA, CA 91125, USA  
*E-mail address:* `cdelaney@caltech.edu`

MATHEMATICS DEPARTMENT, CALTECH, 1200 E. CALIFORNIA BLVD. PASADENA, CA 91125, USA  
*E-mail address:* `matilde@caltech.edu`